REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	14 Nov 1997	Final Progress 1/94 - 12/97

4. TITLE AND SUBTITLE
Formal Models Used for Automation in Software Development

5. FUNDING NUMBERS

6. AUTHOR(S)
Luqi and Valdis Berzins

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Computer Science Department
U.S. Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/ MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Office
P.O. Box 12211
4300 South Miami Blvd.
Research Triangle Park, NC 27709-2211

10. SPONSORING/ MONITORING AGENCY REPORT NUMBER
ARO 30989.33-MA

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This project is investigating formal models that can support automated methods supporting software development. We have focused on automation support for requirements elicitation, particularly for prototyping and the gathering requirements remotely via the Internet; on automation support for software evolution, particularly for automatically detecting the need for software maintenance actions using non-monotonic logic, for capturing requirements dependencies and justifications using the REMAP extension of the IBIS model, for combining several modifications to a system, for coordinating parallel efforts of several designers and automating the associated configuration management tasks, and on automation support for software construction, particularly for using specifications in the design of software architectures, for automated generation of schedules for hard real-time software, and for retrieval of reusable software components.

14. SUBJECT TERMS
Software evolution, requirements elicitation, software reuse, prototyping, change merging, formal models

15. NUMBER OF PAGES
7

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

FORMAL MODELS USED FOR AUTOMATION IN SOFTWARE DEVELOPMENT

Final Progress Report

Luqi and Valdis Berzins
14 November 1997

U.S. Army Research Office
Contract/Grant Number: MIPR7ANPSARO09
(ARO proposal number: 30989-MA)

Naval Postgraduate School

1. Statement of the problem studied:

The objective of this research is the design of an integrated set of formal models and methods for automating a wide range of design and development tasks for real-time systems. The methods we used focus on automation of design activities that appear in an evolutionary prototyping approach to software development. This research used the state-of-the-art formal methods in software engineering to construct a cohesive set of formal models. These models were used to create and to unify automated processes for computer aided prototyping. Mathematical models for implementing a set of automated and integrated software tools were also developed. This research combines very-high-level specification abstractions and concepts with formal real-time models, automated management of software design data and human resources, transformations, change merging, and automated retrieval of reusable software components to provide automated methods for generating real-time programs and for coordinating teams of developers.

2. Summary of the most important results:

This project is investigating formal models that can support automated methods supporting software development. We have focused on automation support for requirements elicitation, particularly for prototyping and the gathering requirements remotely via the Internet; on automation support for software evolution, particularly for automatically detecting the need for software maintenance actions using non-monotonic logic, for capturing requirements dependencies and justifications using the REMAP extension of the IBIS model, for combining several modifications to a system, for coordinating parallel efforts of several designers and automating the associated configuration management tasks; and on automation support for software construction, particularly for using specifications in the design of software architectures, for automated generation of schedules for hard real-time software, and for retrieval of reusable software components.

We have explored applications of non-monotonic logic to software evolution in two different contexts: capturing design rationale and detecting context shifts via inconsistencies. In the first effort, we used extensions of the IBIS model to capture relationships between requirements issues to be resolved via prototyping, possible designed choices, and reactions to prototype demonstrations by representatives of potential user groups for a proposed system. Non-monotonic logic appears to be useful in this context because requirements are often the results of trade-offs between conflicting concerns or negotiations between user groups with different value judgements on particular issues. We also developed models and tools to support the gathering of user input remotely based on the World-wide Web technology.

We have developed a front end to Prolog that realized an answering mechanism corresponding to an extended non-monotonic logic, and have combined it with a simulator for a subset of the prototyping language PSDL. The simulator for the subset of PSDL that incorporates this answering mechanism can monitor the execution of a prototype against assumptions about the system environment. The extended answering mechanism can detect situations where the assumptions about the environment of the proposed system have changed to the point where the previous version of the requirements is no longer completely valid, and an evolution step is needed to bring the requirements back into conformance with reality.

We have developed change-merging methods for software specifications and software architectures. The software specification work treats the black-box specifications expressed in a specification language based on second order logic. The approach integrates a model of interfaces that can support merging changes to module signatures as well as changes to details of module behav-

ior. The work on merging changes to software architectures extends our previous approach based on program slicing for the prototyping language PSDL and an algorithm for merging changes to PSDL programs. The previous method produced a merged design in the form of a single level data flow diagram with annotations. This is not satisfactory for large scale applications because the single level data flow diagrams are too complex for people to understand and use as a basis for further design enhancements. We developed an improved method that combines the corresponding changes to the design hierarchies and uses the result to reconstruct the updated hierarchical structure of the merged design. We also provided a technique for automatically resolving structural conflicts between changes.

Our work on software reuse has resulted in a semantic method for software component search that can simultaneously achieve high precision and high recall, a software architecture for efficiently implementing the method. Partial matches are ranked by semantic closeness. The method uses symbolic test cases in the form of ground equations. The software architecture is based on multi-level filtering approach that uses database indexing and fast rough filters to cut down the number of candidates before the more computationally expensive semantic filters are applied.

3. Publications and technical reports:

(A) Refereed Journal Publications

(1) D. Dampier, Luqi, V. Berzins, "Automated Merging of Software Prototypes", *Journal of Systems Integration*, Vol. 4, No. 1, Feb. 1994, pp. 33-49.

(2) V. Berzins, "Software Merge: Semantics of Combining Changes to Programs", *ACM TOPLAS*, Nov. 1994, pp. 1875-1903.

(3) Luqi, D. Cooke, "How to Combine Nonmonotonic Logic and Rapid Prototyping to Help Maintain Software", *International Journal of Software Engineering and Knowledge Engineering*, Vol. 5, No. 1, March 1995, pp. 89-118.

(4) B. Ramesh, Luqi, "An Intelligent Assistant for Requirements Validation for Embedded Systems", *Journal of Systems Integration*, Vol. 5, No. 2, 1995, pp. 157-177.

(5) Luqi, "System Engineering and Computer-Aided Prototyping", *Journal of Systems Integration*, special issue on Computer Aided Prototyping (Vol. 6, No. 1, 1996), pp. 15-17.

(6) Luqi, M. Shing, "Real-Time Scheduling for Software Prototyping", *Journal of Systems Integration*, special issue on Computer-Aided Prototyping (Vol. 6, No. 1, 1996), pp. 41-72.

(7) J. Goguen, D. Nguyen, J. Meseguer, Luqi, D. Zhang, V. Berzins, "Software Component Search", *Journal of Systems Integration*, special issue on Computer Aided Prototyping (Vol. 6, No. 2, 1996), pp. 93-134.

(8) V. Berzins, D. Dampier, "Software Merge: Combining Changes to Decompositions", *Journal of Systems Integration*, special issue on Computer-Aided Prototyping (Vol. 6, No. 1-2, 1996), pp. 135-150.

(9) V. Berzins, Luqi, "Software Evolution in Prototyping", *Chinese Journal of Advanced Software Research*, Vol. 3, No. 3, (1996), pp. 260-275.

(10) Luqi, J. Goguen, "Formal Methods: Promises and Problems", *IEEE Software*, Vol. 14, No. 1, Jan. 1997, pp. 73-85.

(11) D. Cooke, Luqi, "A Logic-Based Approach to Software Maintenance", to appear in *Annuals of Mathematics and AI.*

(12) V. Berzins, "Recombining Changes to Software Specifications", to appear, *Journal of Systems and Software*, Aug, 1998.

(13) Luqi, C. Chang, H. Zhu, "Specifications in Software Prototyping", to appear in *Journal of Systems and Software*, Aug, 1998.

(B) Conference Publications:

(1) Luqi, M. Shing, "Teaching Hard Real-Time Software Development via Prototyping", *Proceedings of the International Workshop on Software Engineering Education*, at the International Conference on Software Engineering, Sorrento, Italy, May 21, 1994, pp. 199-211.

(2) V. Berzins, "Software Merge: Models and Properties", *Proceedings of the 6th International Conference on Software Engineering and Knowledge Engineering*, Jurmala, Latvia, June 20-23, 1994, pp. 225-232.

(3) Luqi, J. Goguen, "Suggestions for Progress in Software Analysis, Synthesis and Certification", *Proceedings of the 6th International Conference on Software Engineering and Knowledge Engineering*, Jurmala, Latvia, June 20-23, 1994, pp. 501-507.

(4) S. Badr, Luqi, "Automation Support for Concurrent Software Engineering", *Proceedings of the 6th International Conference on Software Engineering and Knowledge Engineering*, Jurmala, Latvia, June 20-23, 1994, pp. 46-53.

(5) Luqi, "Monterey Workshop 94: Software Evolution - Increasing the Practical Impact of Formal Methods in Computer Aided Software Development", *Proceedings of Monterey Workshop 94*, Monterey, CA, Sept. 7-9, 1994, pp. 1-9.

(6) D. Dampier, V. Berzins, "Software Change-Merging in Dynamic Evolution", *Proceedings of Monterey Workshop 94*, Monterey, CA, Sept. 7-9, 1994, pp. 38-41.

(7) S. Badr, V. Berzins, "A Software Evolution Control Model", *Proceedings of Monterey Workshop 94*, Monterey, CA, Sept. 7-9, 1994, pp. 160-171.

(8) Luqi, J. Goguen, V. Berzins, "Formal Support for Software Evolution", *Proceedings of Monterey Workshop 94*, Monterey, CA, Sept. 7-9, 1994, pp. 10-21.

(9) Luqi, H. Yang, X. Zhang, "Constructing an Automated Testing Oracle: An Effort to Produce Reliable Software", *Proceedings of COMPSAC 94*, Taipei, Taiwan, Nov. 7-9, 1994, pp. 228-233.

(10) D. Dampier, V. Berzins, Luqi, M. Shing, D. Dolk, C. Rasmussen, "A Slicing Method for Semantics-Based Change-Merging of Software Prototypes", *Proceedings of Computers in Engineering Symposium*, Houston, Texas, Jan. 29 - Feb. 1, 1995, PD-Vol.67, Software Systems in Engineering, ASME 1995, Edited by D. Cooke et al., pp. 87-92.

(11) Luqi, J. Goguen, V. Berzins, D. Dampier, "Engineering Support for Software Evolution", *Proceedings of Computers in Engineering Symposium*, Houston, Texas, Jan. 29-Feb. 1, 1995. PD-Vol.67, Software Systems in Engineering, ASME 1995, Edited by D. Cooke et. al., pp. 161-171.

(12) D. Dampier, M. Kindl, R. Byrnes, Luqi, "Rapid Prototyping of Army Embedded Software

Systems", *Proceedings of Seventh Annual Software Technology Conference*, Salt Lake City, Utah, April 9-14, 1995.

(13) J. Goguen, Luqi, "Formal Methods and Social Context in Software Development", *Proceedings of the Sixth International Joint Conference on the Theory and Practice of Software Development*, TAPSOFT95, Aarhus, Denmark, May 22-26, 1995, pp. 62-81.

(14) M. Shing, Luqi, "Functional Specification and Prototyping for a Generic C3I Workstation", *Proceedings of 1995 Symposium on Command and Control Research and Technology*, Washington, DC, June 19-23, 1995, pp. 119-131.

(15) V. Berzins, Luqi, M. Shing, "Computer Aided Prototyping System", *Proceedings of 1995 International Conference on Software Engineering and Data Engineering*, Washington, DC, June 22-24, 1995, p. 499.

(16) Luqi, V. Berzins, "Logic, Software Engineering and Prototyping", *Proceedings of the Logic and Software Engineering Workshop*, Software Institute, National Academy of Science, Beijing, China, Aug. 15, 1995.

(17) V. Berzins, Luqi, "Software Evolution in Prototyping", *Proceedings of the Logic and Software Engineering Workshop, Software Institute*, National Academy of Science, Beijing, China, Aug. 15, 1995.

(18) D. Zhang, Luqi, "Formal Analysis of Inconsistency and Redundancy in Knowledge Bases", *Proceedings of IJCAI'95 Workshop on Validation & Verification of Knowledge Based Systems*, Montreal, Quebec, Canada, Aug. 19, 1995, pp. 110-116.

(19) Luqi, V. Berzins, "Software Architecture in Computer-Aided Prototyping", *Proceedings of 1995 Monterey Workshop on Increasing the Practical Impact of Formal Methods in Computer Aided Software Development: Software Architecture*, Monterey, CA, Sep. 12-14, 1995, pp. 44-57.

(20) Luqi, "Specification Based Software Architecture", *Proceedings of 1995 Monterey Workshop on Increasing the Practical Impact of Formal Methods in Computer Aided Software Development: Software Architecture*, Monterey, CA, Sep. 12-14, 1995, pp. 1-4.

(21) V. Berzins, M. Shing, "Summary of 95 Monterey Workshop: Specification-Based Software Architectures", *Proceedings of 1995 Monterey Workshop on Increasing the Practical Impact of Formal Methods in Computer Aided Software Development: Software Architecture*, Monterey, CA, Sep. 12-14, 1995, pp. 107-112.

(22) M. Shing, V. Berzins, Luqi, "Computer Aided Prototyping System (CAPS)", *Proceedings of the Software Technology Conference*, Salt Lake City, Utah, April 21-26, 1996, pp. 462-463.

(23) V. Berzins, "Recombining Changes to Software Specifications", *Proceedings of the 8th International Conference on Software Engineering and Knowledge Engineering*, Lake Tahoe, CA, June 10-12, 1996, pp. 136-144.

(24) Luqi, "Specifications in Software Prototyping", *Proceedings of the 8th International Conference on Software Engineering and Knowledge Engineering*, Lake Tahoe, CA, June 10-12, 1996, pp. 189-197.

(25) Luqi, V. Berzins, D. Nguyen, D. Zhang, "Multi-Level Filtering for Software Component Retrieval", *Proceedings of the 1996 International Conference on Circuits and System Sci-*

ences, Shanghai, China, June 20-25, 1996, pp. 284-287.

(26) D. Rusin, Luqi, M. Scanlon "SIDS Wireless Acoustic Monitor (SWAM)", *Proceedings of the 21st Int. Conf. on Lung Sounds*, The International Lung Sounds Association, Chester, England, Sep 4-6, 1996.

(27) V. Berzins, Luqi, M. Shing, "Scheduling Real-Time Software Prototypes", *Proceedings of the 2nd International Symposium on Operations Research and it s Application*, Guilin, China, December 11-13, 1996, pp. 614-623.

(28) V. Berzins, O. Ibrahim, Luqi, "A Requirements Evolution Model for Computer Aided Prototyping", Proceedings of the 9th International Conference on Software Engineering and Knowledge Engineering, Madrid, Spain, June 17-20, 1997, pp. 38-47.

(29) T. Leonard, V. Berzins, Luqi, M. Holden, "Gathering Requirements from Remote Users", *Proceedings 9th International Conference on Tools with Artificial Intelligence*, Newport Beach, CA, November 3-8, 1997, pp. 462-471.

4. Scientific personnel supported by this project: Luqi, Valdis Berzins

5. Report of inventions: None.

Appendix- abstracts of papers published in 1997

A Requirements Evolution Model for Computer Aided Prototyping

This paper presents a model for requirements evolution and analysis in the context of iterative prototyping of large embedded real-time systems. This model captures user reactions to demonstrated behavior of the prototype and maps these reactions into requirements changes. The model provides the basis for automated support for requirements evolution and validation. This paper explores how a request of a change can be derived form the justifiable user responses to the demonstrated behavior of the prototype.

Gathering Requirements from Remote Users

We describe a distributed requirements engineering environment using computer aided software engineering tools linked together through the Internet. We created this distributed requirements engineering environment using Microsoft's Personal Web Server (PWS), Microsoft's Open Database Connectivity (ODBC) technology, Netscape Communicator, Microsoft's Internet Explorer, Microsoft's Access97 database, and a set of PERL scripts that are executed by users of the environment to perform database operations. We show how we added basic security features to the Internet accessible database.

Recombining Changes to Software Specifications

This paper proposes a model of software changes for supporting the evolution of software prototypes. We decompose software evolution steps into primitive substeps that correspond to monotonic specification changes. This structure is used to rearrange chronological derivation sequences into idealized conceptual derivation structures containing only meaning-extending changes, and to automatically combine different changes to a specification. A set of example illustrates the ideas.

A Logic-Based Approach to Software Maintenance

This paper provides an overview of the relationship between recent work in logic programming and recent developments in software engineering. The relationship to software engineering is more specifically concerned with how formal specifications can be used to explain and represent the basis of software maintenance and evolution.